

# MICROSERVICIOS



# Temario

- › Maven
- › Springboot
- › Patrones de diseño
- › Api Design
- › Orquestación
- › Docker
- › Kubernetes
- › AWS ECS



# HELLO!

**Oscar Garcia**

Arquitecto de software

<https://github.com/ogarules/CursoMicro2020>

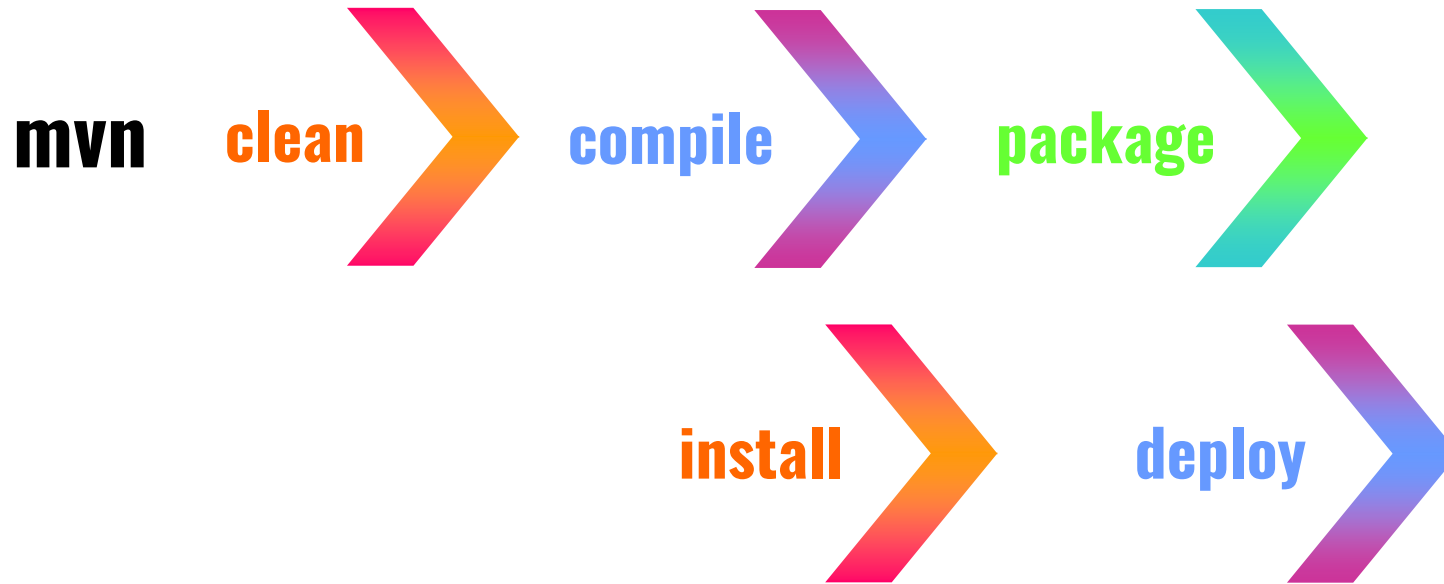
# 1. Maven

Gestión y construcción de proyectos

***“Herramienta de software para la gestión  
y construcción de proyectos Java.*”**

***Utiliza un Project Object Model (POM) para  
describir el proyecto de software a  
construir, sus dependencias de otros  
módulos y componentes externos, y el  
orden de construcción de los elementos.”***

# Lifecycle



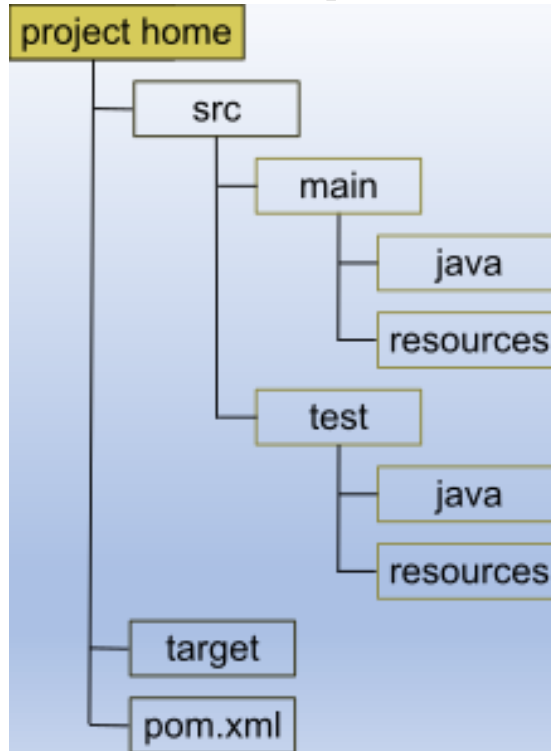
# Descargar maven

**<https://maven.apache.org/download.cgi>**

Verificar instalación con:

**mvn -v**

# Estructura proyecto maven



**src/main/java**  
**src/main/groovy**  
**src/main/scala**  
**src/test/java**  
**src/test/groovy**  
**src/test/scala**



# El archivo pom.xml

## Información del proyecto

groupId [package]  
(com.junit, com.example)  
artifactId [Nombre de la aplicación]  
version [Version de la aplicación] (snapshot / releases)  
packaging [Tipo de empaquetado] (jar, war, ear)

## Dependencias

Dependencias utilizadas por la aplicación

## Build

Plugins  
Estructura del directorio

## Repositories

Repositorios de donde se obtienen las dependencias

# El archivo pom.xml

## Dependencias [Scopes]

compile [Artefactos disponibles en todo momento]

provided [Disponible en cualquier lugar, pero no se incluye en el empaquetado final]

runtime [No es necesaria para compilación, pero si para tiempo de ejecución]

test [Solo disponible durante la compilación de los test]

system [Especifica que el jar se encuentra en un path físico del sistema]

import [dependencyManagement, compartir package entre diferentes poms]

# El archivo pom.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5      <modelVersion>4.0.0</modelVersion>
6
7      <groupId>com.some.company</groupId>
8      <artifactId>some-project</artifactId>
9      <version>0.0.1</version>
10
11     <name>some-project</name>
12     <!-- FIXME change it to the project's website -->
13     <url>http://www.example.com</url>
14
15     <properties>
16         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17         <maven.compiler.source>1.7</maven.compiler.source>
18         <maven.compiler.target>1.7</maven.compiler.target>
19     </properties>
20
21     <dependencies>
22         <dependency>
23             <groupId>junit</groupId>
24             <artifactId>junit</artifactId>
25             <version>4.11</version>
26             <scope>test</scope>
27         </dependency>
28     </dependencies>
29
30 </project>

```

# Lab 1 Generar proyecto maven

## Descargar visual studio code

<https://code.visualstudio.com/download>

## Genera estructura de directorios maven

## Generar una clase main y una prueba unitaria

## Ejecutar

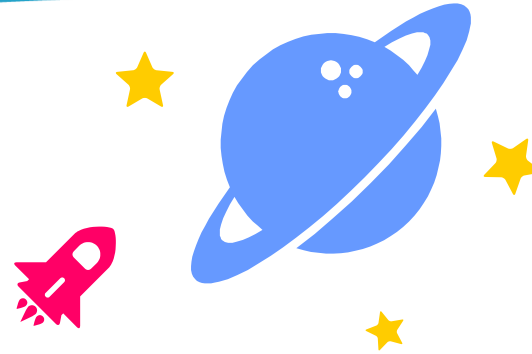
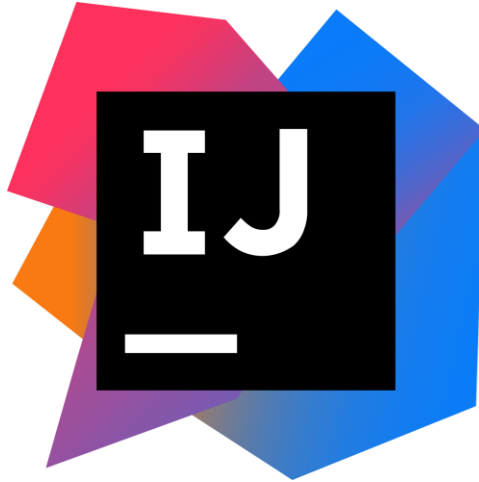
`mvn clean compile package`

`mvn clean test`

`mvn clean package`

`mvn clean`

# Lab 2



Maven en IntelliJ

# 2. Springboot

Framework, IoC, opensource

***“Framework de desarrollo de  
aplicaciones basado en spring core  
Puede desplegarse en modo standalone  
Fortaleza inyección de dependencias”***

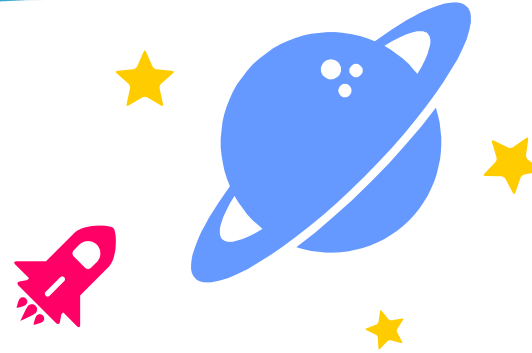
# Springboot



***jetty://***



# Lab 3



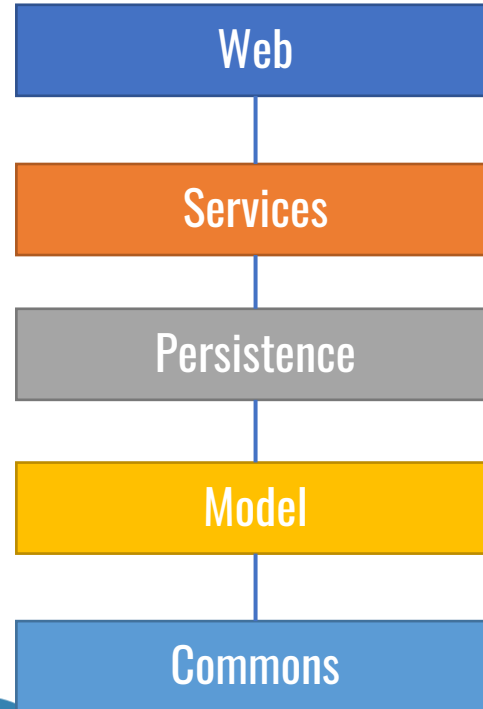
Springboot

<https://github.com/IngJavierR/SpringbootCourse>

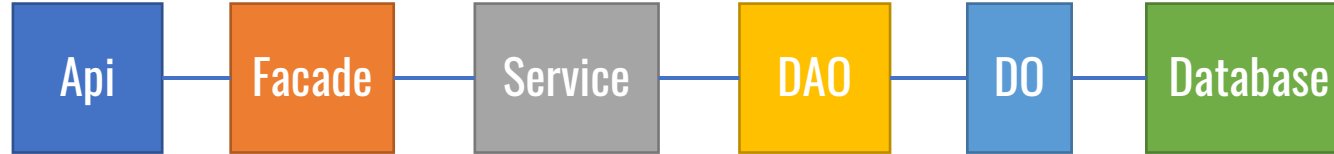
# 3. Patrones de diseño

Reusabilidad en desarrollo de software

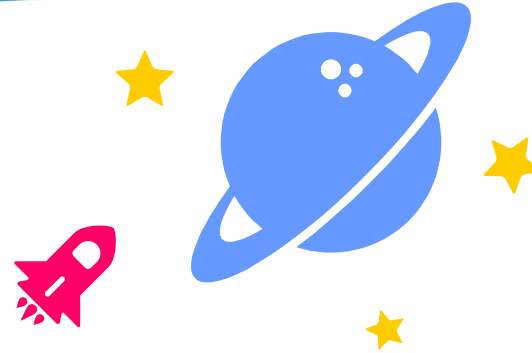
# Estructura maven multimodulo



# Patrones utilizados



# Lab 4



Generando servicios

# 4. Api Design

Mejores prácticas

# Api design



# Api design

Usar sustantivos y evitar los verbos

`/dogs` `/dogs/1234`

Evitar

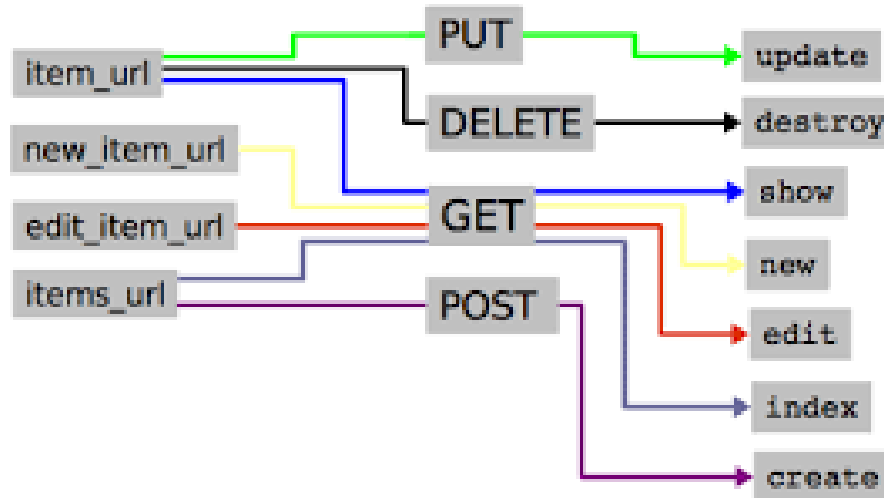
`/getDogs` `/getDogs/1234`





# Api design

Utilizar verbos http correctos



# Api design

Recurso	POST	GET	PUT	DELETE
/dogs	Crea un nuevo perro	Lista de perros	Actualiza lista de perros	Borra todos los perros
/dogs/1234	Bad hombres	Muestra perro 1234	Si existe actualiza 1234	Borra 1234
			Si no existe error	



# Api design

## Plurales y nombres concretos

GET /dogs?color=red&state=running&location=park

POST /dog

GET /dogs

GET /dog

GET /getAllDogs



# Api design

## HTTP Status Codes

### Level 200 (Success)

200 : OK

201 : Created

203 : Non-Authoritative  
Information

204 : No Content

### Level 400

400 : Bad Request

401 : Unauthorized

403 : Forbidden

404 : Not Found

409 : Conflict

### Level 500

500 : Internal Server Error

503 : Service Unavailable

501 : Not Implemented

504 : Gateway Timeout

599 : Network timeout

502 : Bad Gateway

# Api design

Manejor de errores y logging

Programación orientada a aspectos **AspectJ**

**Aspects:** se definen como "envoltorios" de código. Se parecen a las clases de Java.

Un aspect puede interceptar a las clases

# Api design

## Versionamiento

Twilio /2010-04-01/Accounts/

Salesforce.com /services/data/v20.0/subjects/Account

Facebook ?v=1.0

## Best practice

/v1/accounts

/v2/accounts

## Content-Type

dogs/1 Content-Type: application/json

dogs/1 Content-Type: application/xml

dogs/1 Content-Type: application/png

# Api design

## Respuestas parciales

LinkedIn /people:(id,first-name,last-name,industry)

Facebook /joe.smith/friends?fields=id,name,picture

Google ?fields=title,media:group(media:thumbnail)

/dogs?fields=name,color,location

## Paginación

/dogs?limit=25&offset=50

Facebook - offset 50 and limit 25

Twitter - page 3 and rpp 25 (records per page)

LinkedIn - start 50 and count 25

# Api design

## Busqueda

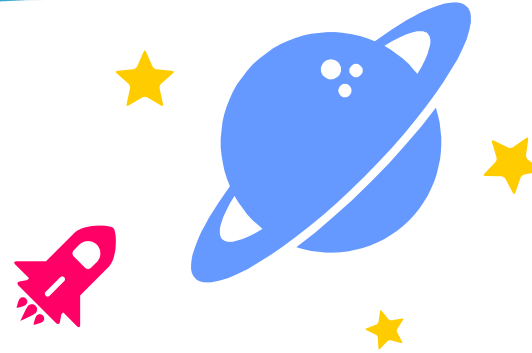
Global search /search?q=fluffy+fur  
/owners/5678/dogs?q=fluffy+fur

## Seguridad

PayPal Permissions Service API  
Facebook OAuth 2.0  
Twitter OAuth 1.0a



# Lab 5



Generando un api

# Lab 5

Operaciones crud de  
usuarios

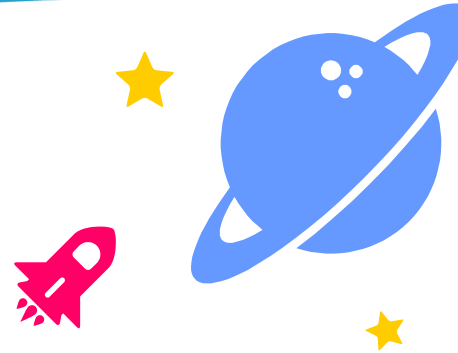
Consultar

Guardar

Actualizar

Borrar

Pruebas unitarias



# RequestMapping

**@PathVariable** - <http://localhost:8080/spring-rest/dog/1>

@RequestMapping(value = "/dog/{id}", method = GET)

@PathVariable("id") long id

**@RequestParam** - <http://localhost:8080/spring-rest/dogs?id=100>

@RequestMapping(value = "/dogs", method = GET)

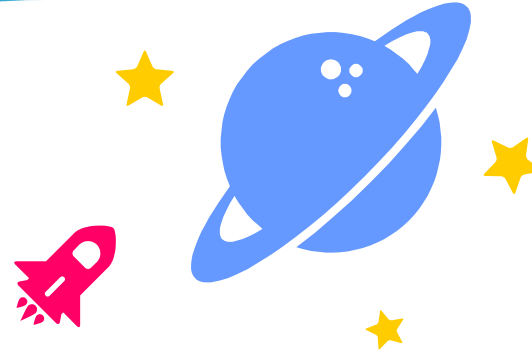
@RequestParam("id") long id

**@RequestBody** - <http://localhost:8080/spring-rest/dog>

@RequestMapping(value = "/dog", method = RequestMethod.POST)

@RequestBody DogTO dog

# Lab 6



@RequestMapping

# Pagination and sort

**@PathVariable** - <http://localhost:8080/spring-rest/dogs/1>

@RequestMapping(value = "/dogs/{id}", method = GET)

@PathVariable("id") long id

**@RequestParam** - <http://localhost:8080/spring-rest/dogs?id=100>

@RequestMapping(value = "/dogs", method = GET)

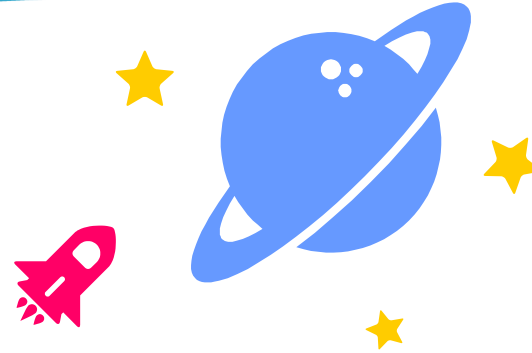
@RequestParam("id") long id

**@RequestBody**

@RequestMapping(value = "/persistPerson", method = RequestMethod.POST)

@RequestBody PersonDTO person

# Lab 7



@Paginator and Sort

# 5. Orquestador

Gestión de microservicios

***“En un ambiente de microservicios los dominios permanecen lo mas separado posible, sin embargo esta separación hace complicado la gestión de los mismos, un orquestador mejorará la administración y el control de las apis involucradas”***



# Documentación



# Documentación

## Que es documentar un API?

Documento técnico que no que puede ser distribuido a los clientes que van a consumir nuestras APIs, la documentación debe ser capaz de transmitir claramente las entradas y salidas que nuestro manejará.

```

http://127.0.0.1:30103/wsdl
Raw XML
<?xml version='1.0' encoding='UTF-8'>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header>
    <os:OS_Header>123</os:OS_Header>
  </soapenv:Header>
  <soapenv:Body>
    <os:CREATE-SERVICE>
      <os:MODIFIER>123</os:MODIFIER>
      <os:object>
        <os:ORDERID>12345</os:ORDERID>
        <os:PONPortAlias>AK</os:PONPortAlias>
        <os:ONTSN>3456</os:ONTSN>
      </os:object>
      <os:Session?</os:Session>
      <!--Optional:-->
      <os:objectParas>
        <os:FDN>123</os:FDN>
        <os:CustomerName>Wiknesh</os:CustomerName>
        <os:ServiceType>FITH</os:ServiceType>
        <!--Optional:-->
        <os:SIPUSERNAME?</os:SIPUSERNAME>
        <!--Optional:-->
        <os:SIPUSERPWD?</os:SIPUSERPWD>
        <!--Optional:-->
        <os:WANPPPUSERNAME?</os:WANPPPUSERNAME>
        <!--Optional:-->
        <os:WANPPPUSERPWD?</os:WANPPPUSERPWD>
      </os:objectParas>
    </os:CREATE-SERVICE>
  </soapenv:Body>
</soapenv:Envelope>
Auth Headers (0) Attachments (0) WS-A WS-RM JMS Headers JMS Property (0) Headers (0) Attac
Error getting response; org.apache.http.conn.HttpHostConnectException: Connection to http://127.0.0.1:30103 refused
  
```

# Documentación

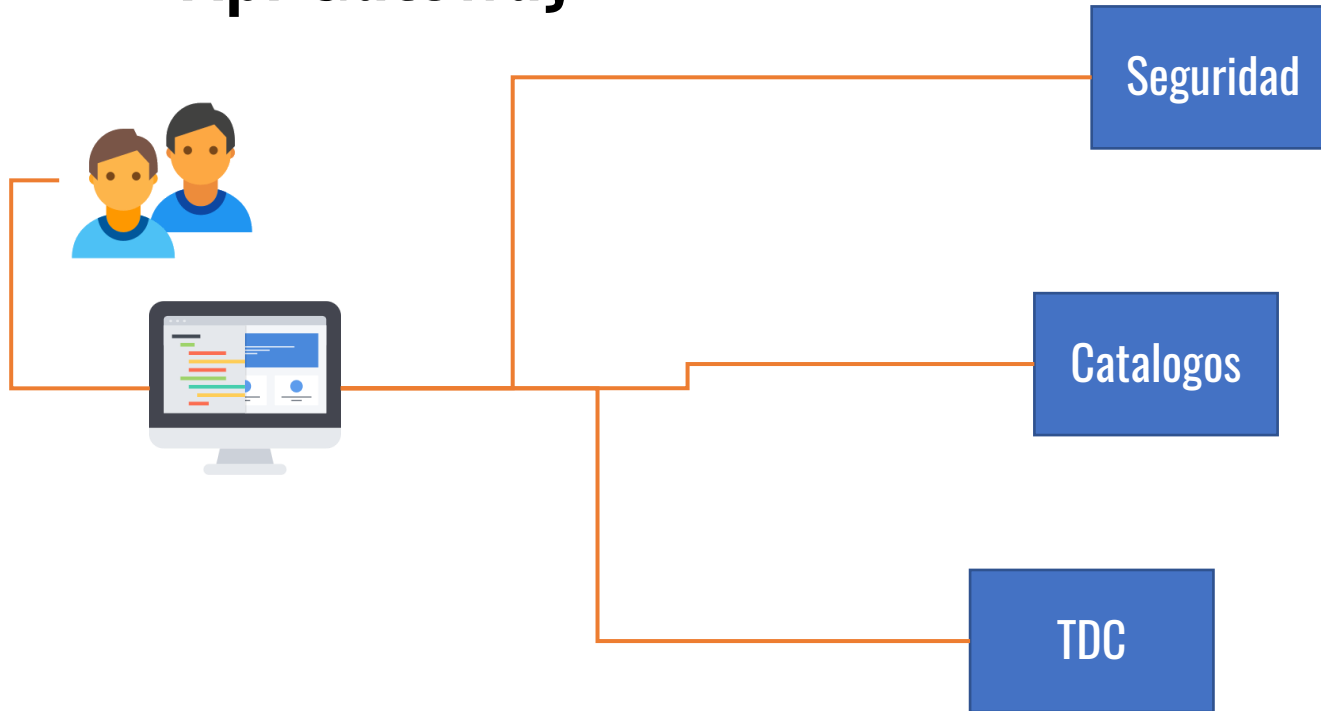
## Importa?

Los servicios rest no generan un contrato natural de entradas y salidas como lo hace un servicio SOAP con el wsdl.

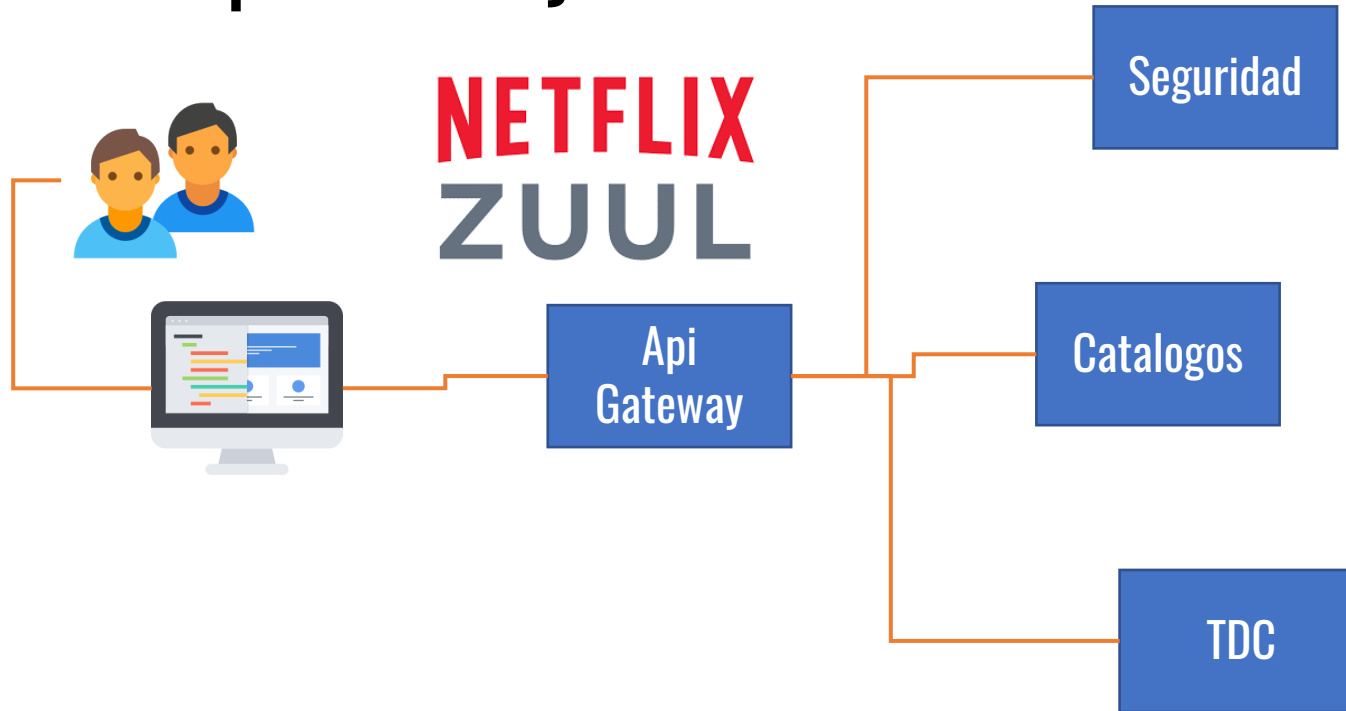


<http://localhost:8090/swagger-ui.html>.

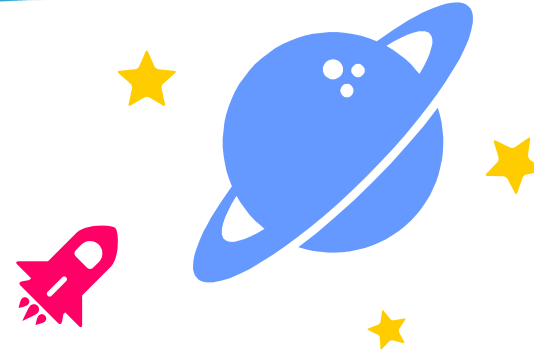
# Api Gateway



# Api Gateway



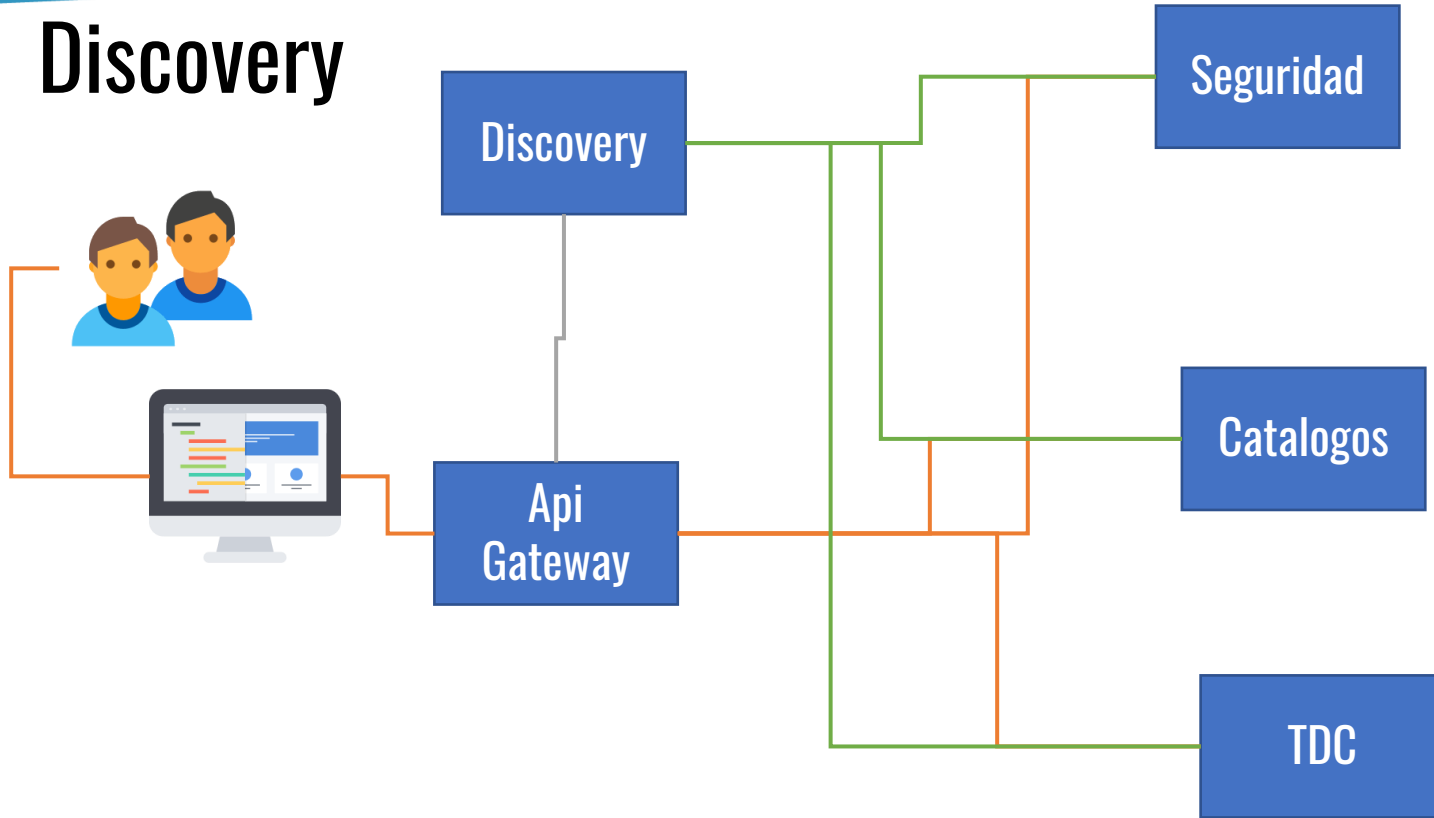
# Lab 8



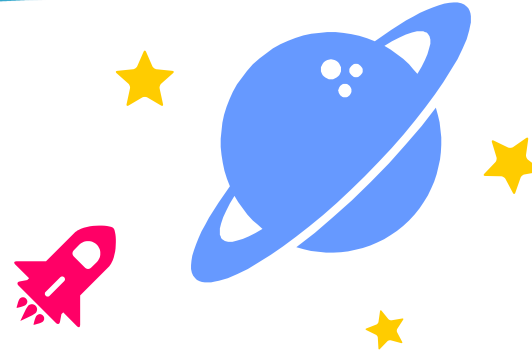
**NETFLIX**  
**OSS**

Netflix OSS Zuul

# Discovery



# Lab 9

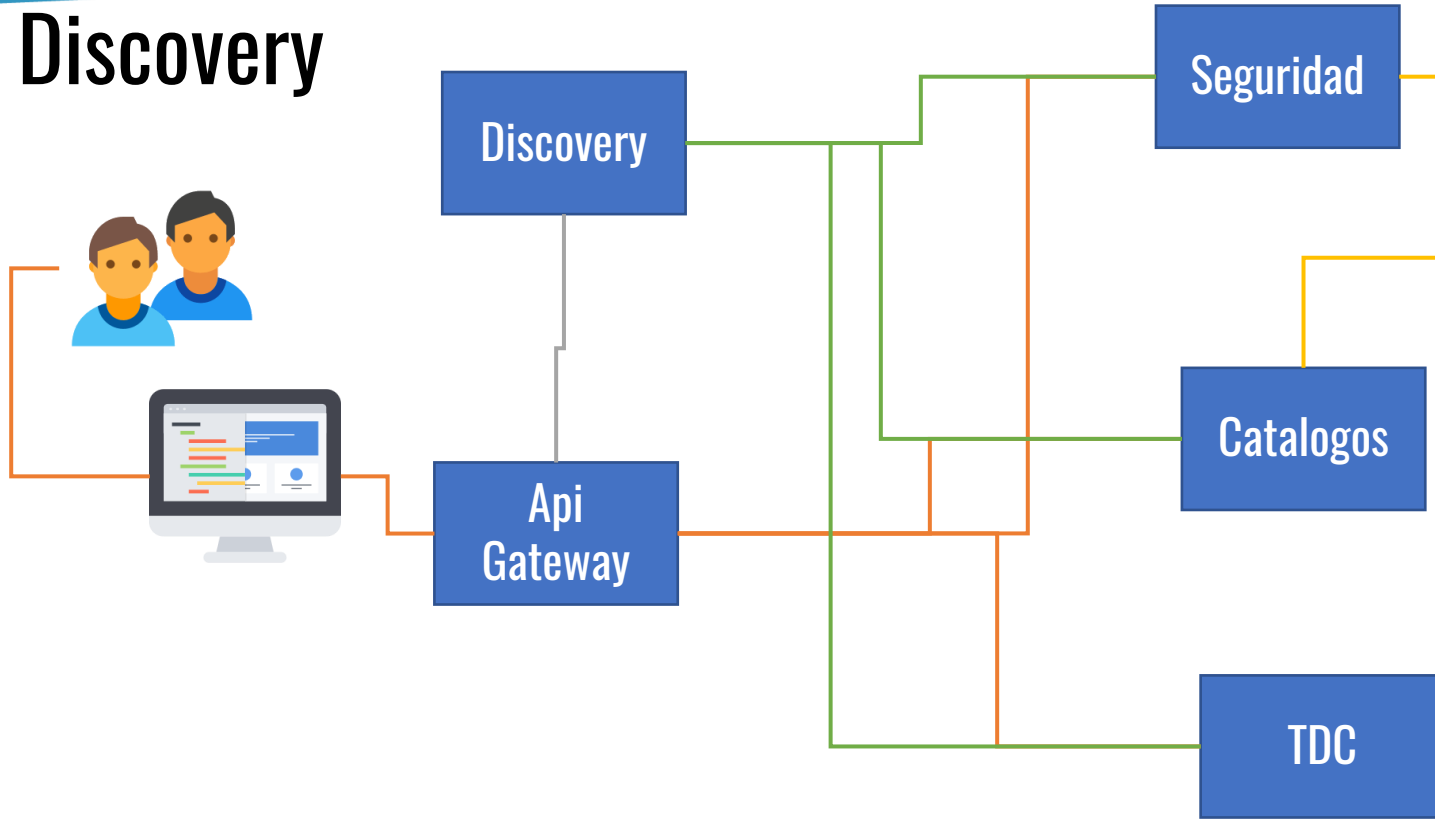


**NETFLIX**  
**OSS**

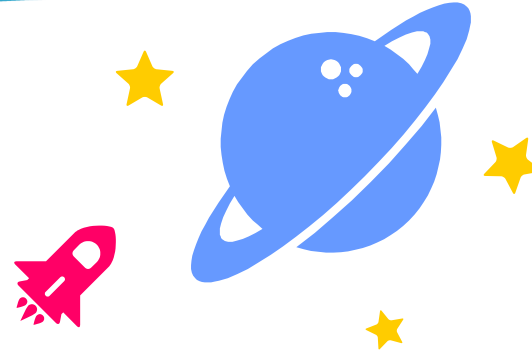
Netflix OSS Eureka



# Discovery



# Lab 10

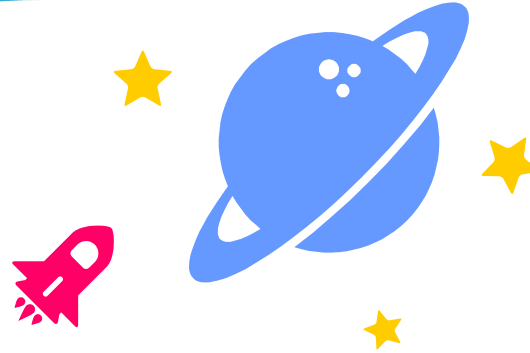


**NETFLIX**  
**OSS**

Comunicación entre microservicios

# Ejercicio

**NETFLIX**  
**OSS**



Generar un sistema con Netflix OSS de comunicación  
entre dos microservicios

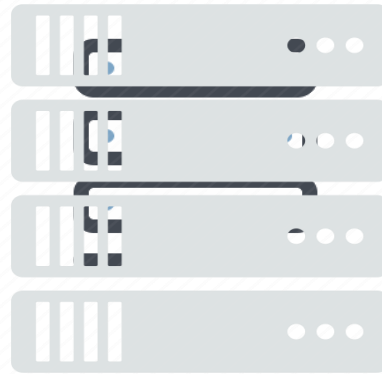
# 6. Contenedores

Automatizar despliegue de aplicaciones

# Tradicional

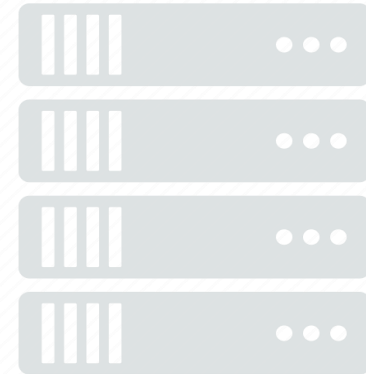


# Tradicional



**Qué tipo de  
servidor se  
requiere para  
esta  
aplicación?**

# Tradicional



vmware®



Microsoft  
Hyper-V

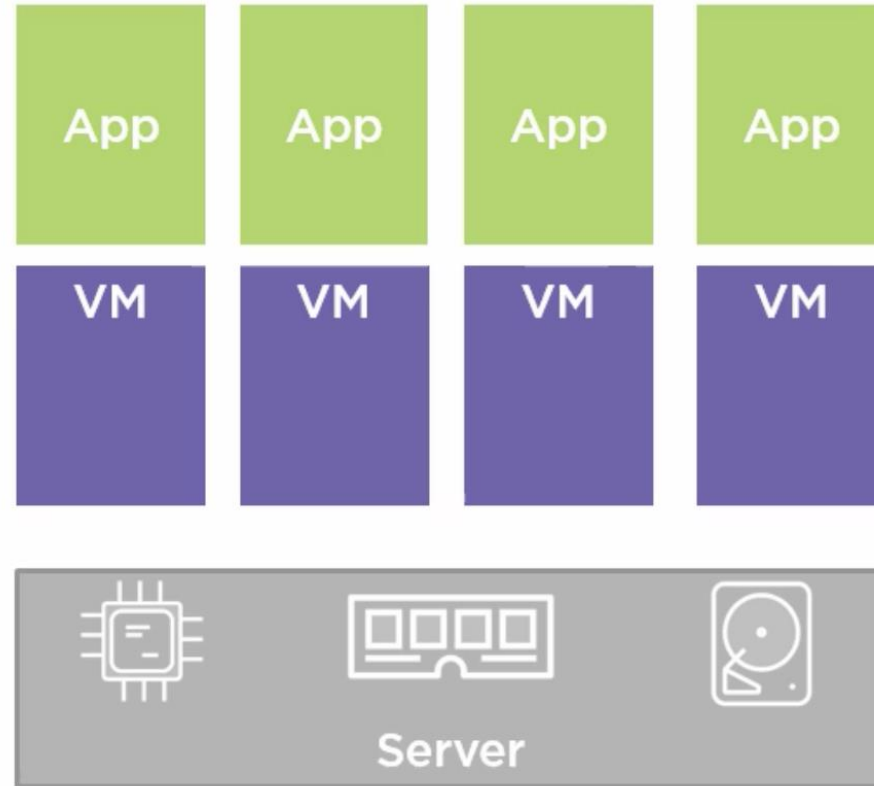


# Tradicional

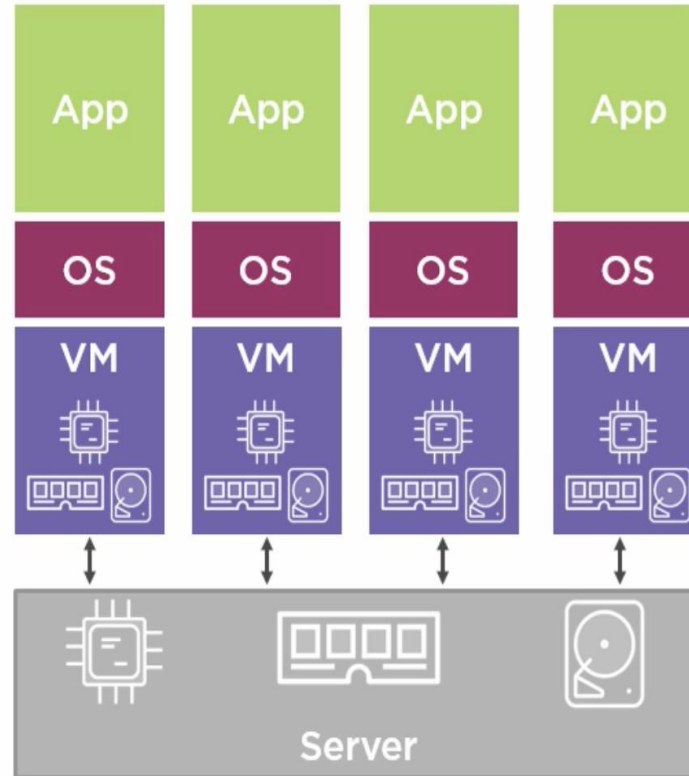




# El problema



# El problema



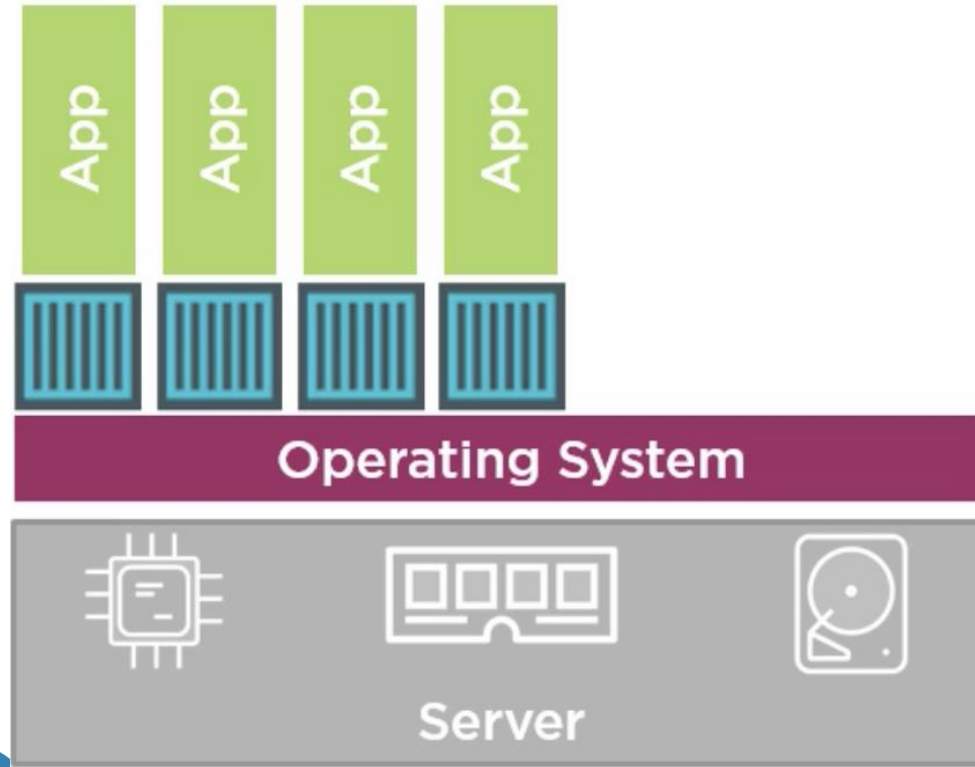
# Contenedores



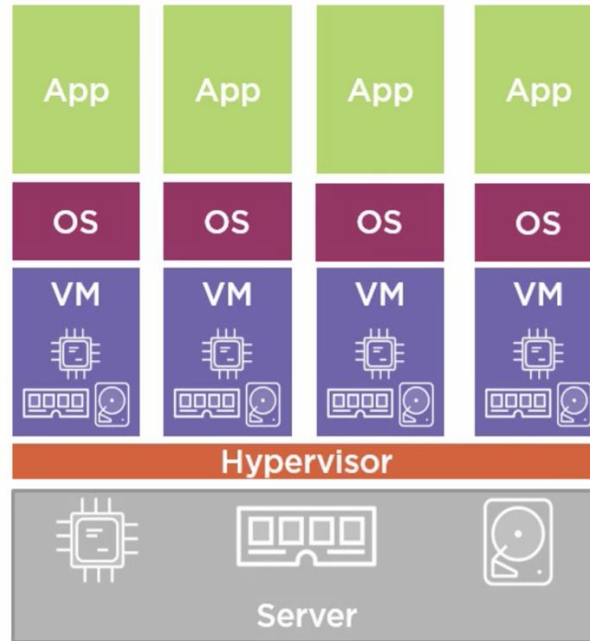
# Contendores



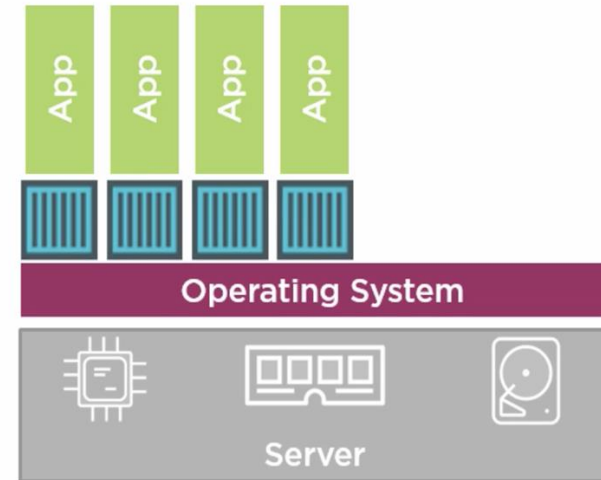
# Contendores



# Contenedores



Hypervisor  
Architecture



Container  
Architecture

# Qué es Docker?

## Docker Company

Secure | <https://www.docker.com/company>



## Open Container Initiative

Secure | <https://www.opencontainers.org>

THE **LINUX** FOUNDATION PROJECTS

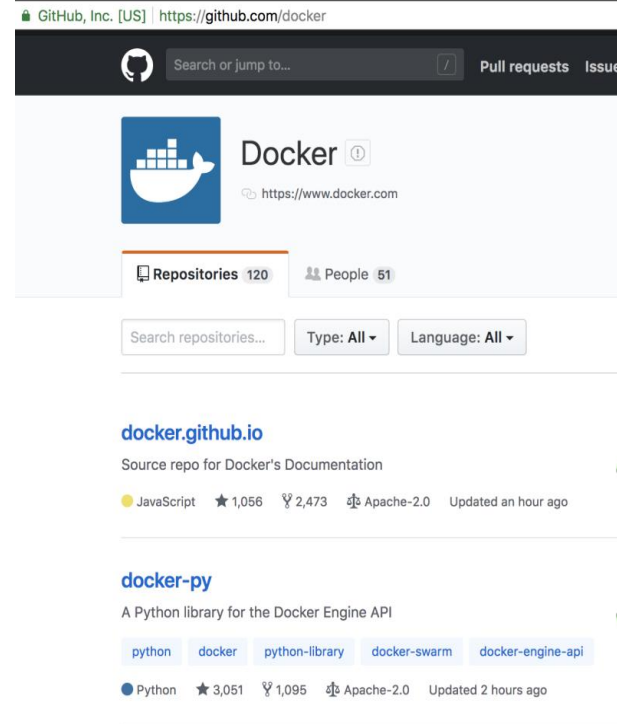


**OPEN** CONTAINER INITIATIVE

About Community Jo



## Docker Project



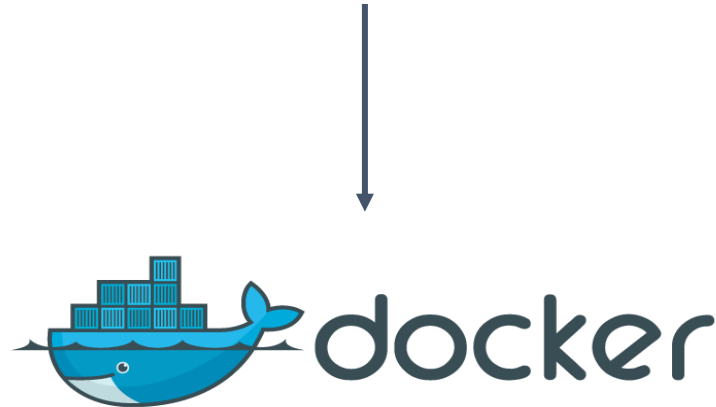


# Docker company

Mayor sponsor del proyecto Docker  
Inicio ofreciendo SaaS  
Proyecto interno con contenedores



dock worker









# Docker project

GitHub, Inc. [US] | <https://github.com/docker>

Search or jump to... Pull requests Issues

 **Docker**   
<https://www.docker.com>

 **Repositories** 120  **People** 51

Search repositories... Type: All Language: All

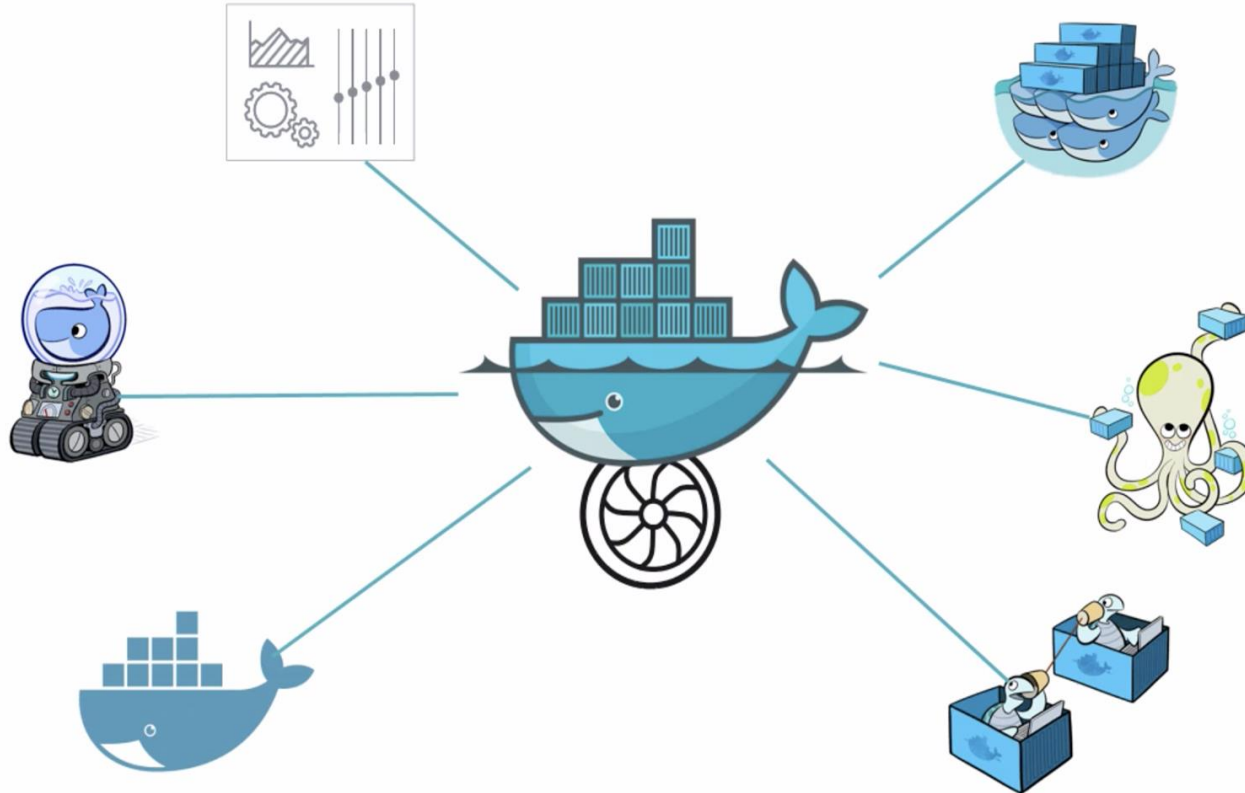
**docker.github.io**  
Source repo for Docker's Documentation  
JavaScript ★ 1,056 2,473 Apache-2.0 Updated an hour ago

**docker-py**  
A Python library for the Docker Engine API  
python docker python-library docker-swarm docker-engine-api  
Python ★ 3,051 1,095 Apache-2.0 Updated 2 hours ago



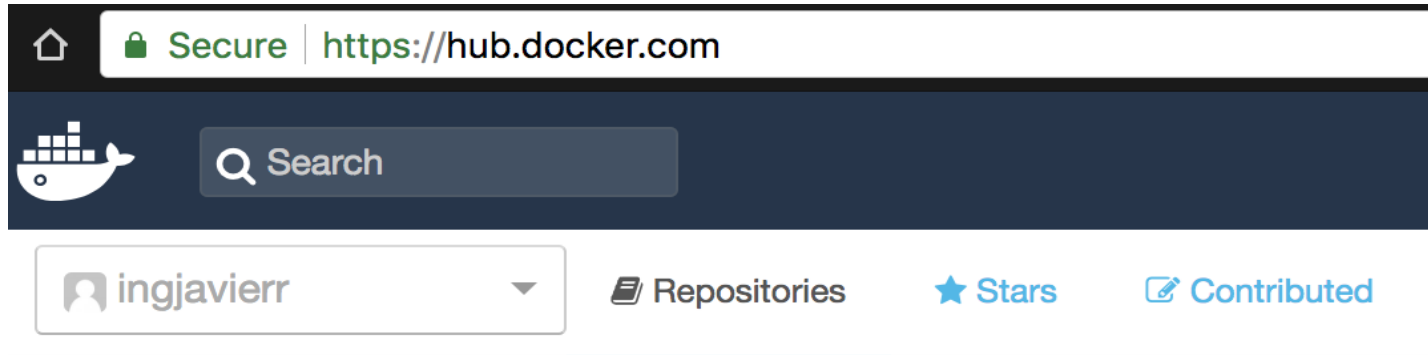
Go

# Docker project



# Docker project

## Docker Hub



Repositories

# Open container initiative



VS



# Open container initiative

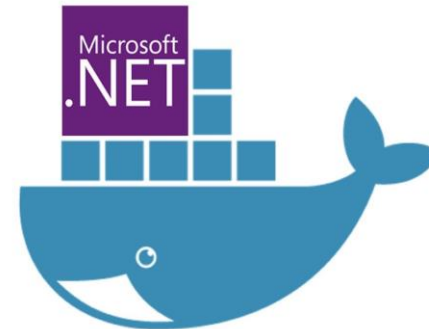


# Que es un contenedor?



Es una instancia de un servidor linux en su mínima expresión.

# Qué puede vivir en contenedor?



# Helloworld



## **docker run hello-world**



# Corriendo mi primer imagen

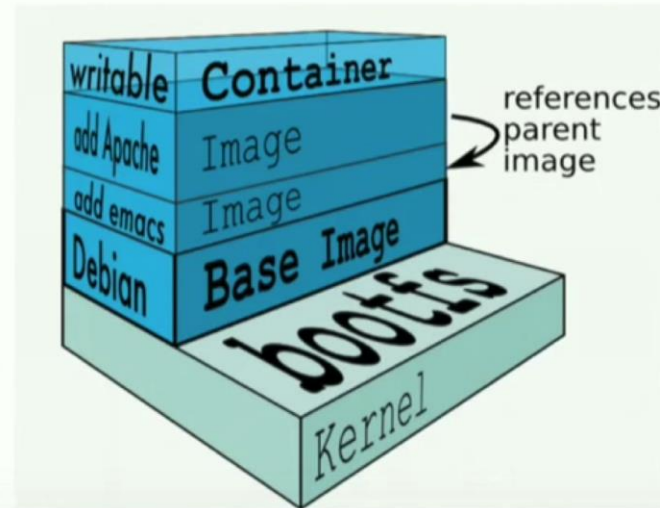
**docker run -d -p "8100:80" ingjavierr/nginx-angular6**

Donde:

- run (Correr la imagen, si no la encuentra local, la descarga)
- -d (Detached mode)
- -p (Puerto Exterior : Puerto interior)
- ingjavierr/nginx-angular6 (Nombre de la imagen)

# Docker image layers

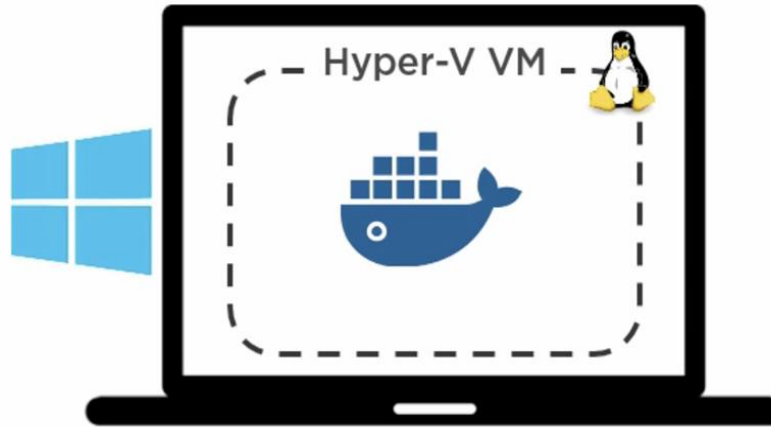
Image layers



# Docker en windows

## Pre-reqs

- Windows 10
- 64-bit
- Clean (ish) install



## Use cases

- Test
- Dev
- ~~Production~~

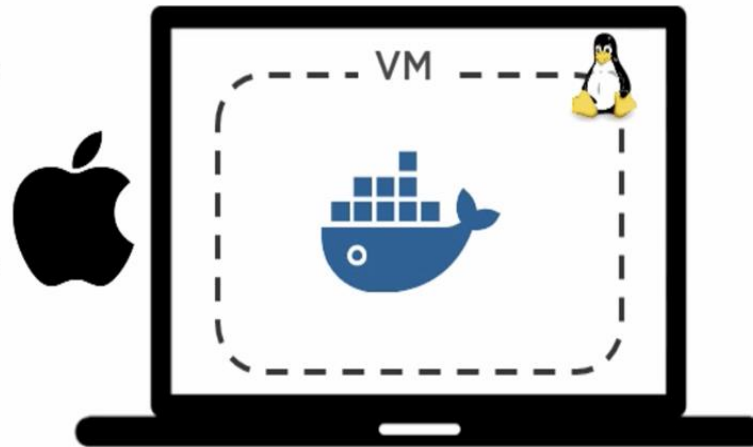
# Docker en MAC

HyperKit

- <https://github.com/docker/hyperkit>

DataKit

- <https://github.com/docker/datakit>



Moby Linux

- Based on Alpine Linux

# Docker en linux (docker version)

```
Client:
 Version:      18.04.0-ce
 API version:  1.37
 Go version:   go1.9.4
 Git commit:   3d479c0
 Built: Tue Apr 10 18:21:36 2018
 OS/Arch:     linux/amd64
 Experimental: false
 Orchestrator: swarm

Server:
 Engine:
  Version:      18.04.0-ce
  API version:  1.37 (minimum version 1.12)
  Go version:   go1.9.4
  Git commit:   3d479c0
  Built:       Tue Apr 10 18:25:25 2018
  OS/Arch:     linux/amd64
  Experimental: false
```

# Docker en windows server 2016+

```
PS C:\windows\system32> docker version
Client:
 Version:      1.12.0-dev
 API version:  1.24
 Go version:   go1.5.3
 Git commit:   8e92415
 Built:        Thu May 26 17:08:34 2016
 OS/Arch:      windows/amd64

Server:
 Version:      1.12.0-dev
 API version:  1.24
 Go version:   go1.5.3
 Git commit:   8e92415
 Built:        Thu May 26 17:08:34 2016
 OS/Arch:      windows/amd64
```

# Linux minificado

```
$ docker run -it -p "8090:80" ubuntu bin/bash
```

```
$ apt-get update -y
```

```
$ apt-get install -y apache2 apache2-utils
```

```
$ apt-get install -y vim
```

```
# /usr/sbin/apache2ctl start
```

```
# vi /var/www/html/hola.html
```

# Dockerfile

Docker puede crear imágenes tomando instrucciones de un archivo de configuración llamado Dockerfile

**FROM ubuntu**

**RUN apt-get update -y && apt-get install -y apache2 apache2-utils**

**COPY hola.html /var/www/html/hola.html**

**CMD /usr/sbin/apache2ctl -D FOREGROUND**

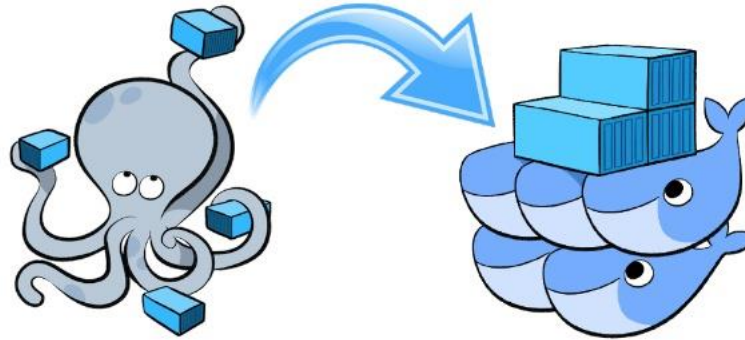
**docker build -t hola .**

**docker run -d -p "8090:80" hola**



# Docker compose

## Docker Compose



# Y también...



# Docker y android



# 7. Kubernetes

Administrar contenedores

***“En un ambiente de contenedores la administración puede hacerse compleja, k8s automatiza despliegues, y facilita la administración las instancias”***

# Historia



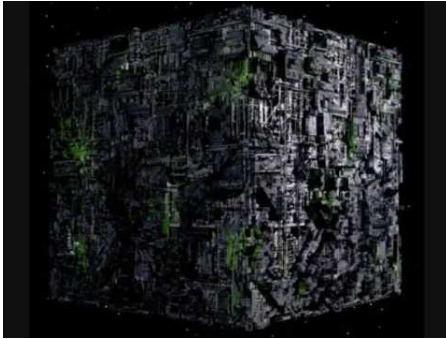
**Donado a la Linux Foundation en 2014**

**Desarrollado en Go**

**<https://github.com/kubernetes>**

**Kubernetes (k8s)**

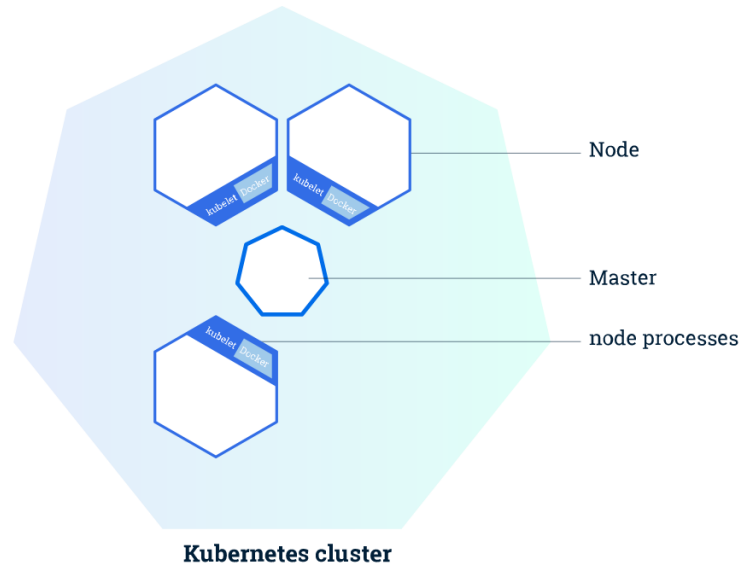
# Historia



**Kubernetes - Helmsman (Timón)**

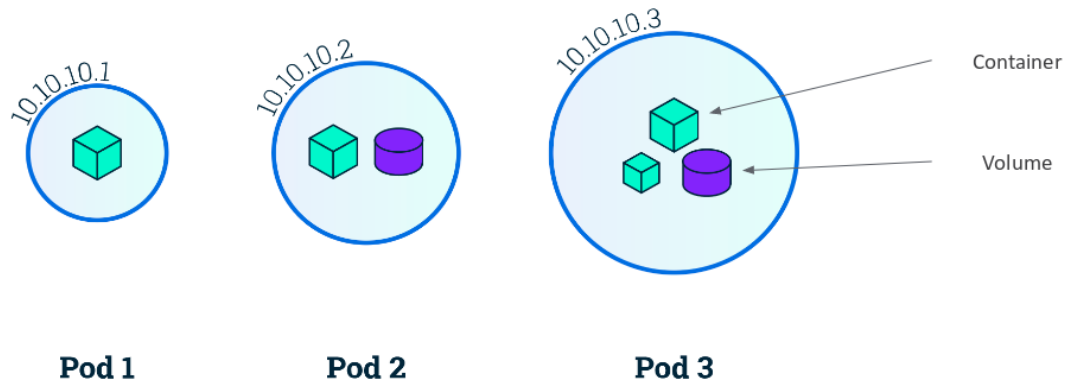
# Arquitectura

## Cluster Diagram

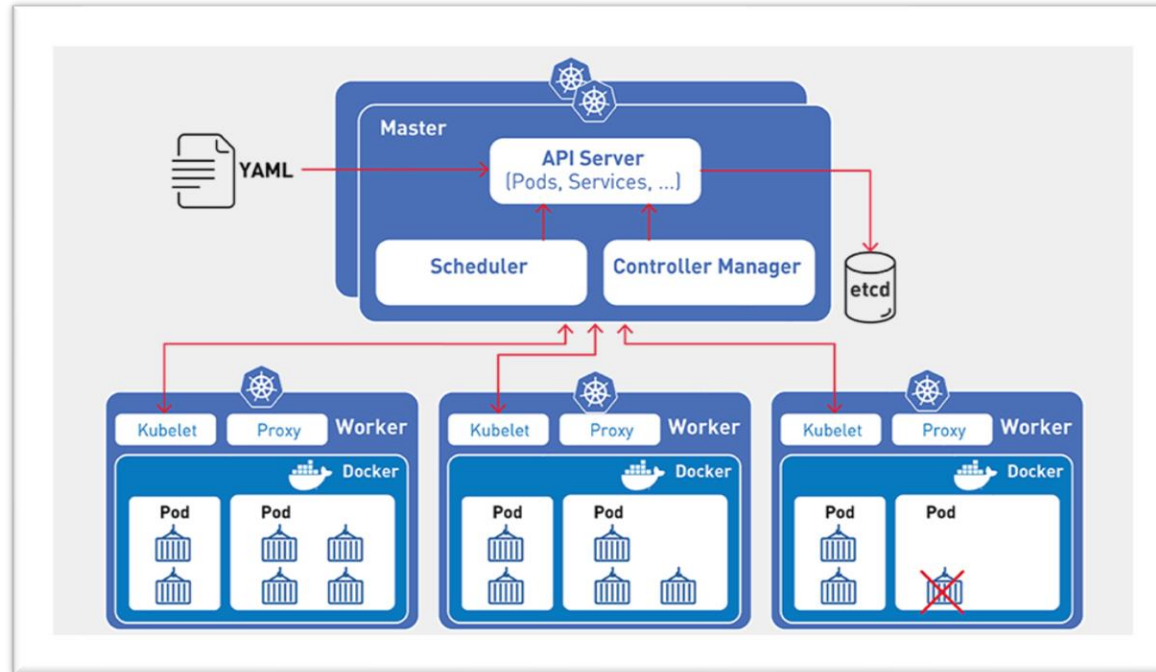




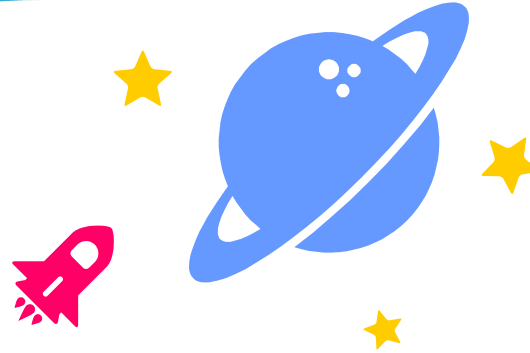
# Arquitectura



# Arquitectura



# Lab 11



Instalar kubernetes (minikube)

# Minikube

## Instalación

<https://kubernetes.io/docs/tasks/tools/install-kubect/>

- > \$ brew install kubectl
- > \$ brew cask install minikube
- > \$ brew install docker-machine-driver-xhyve
- > \$ minikube start --vm-driver=xhyve
- > \$ minikube start --vm-driver=hyperv
- > \$ kubectl config current-context
- > \$ kubectl cluster-info
- > \$ kubectl get nodes

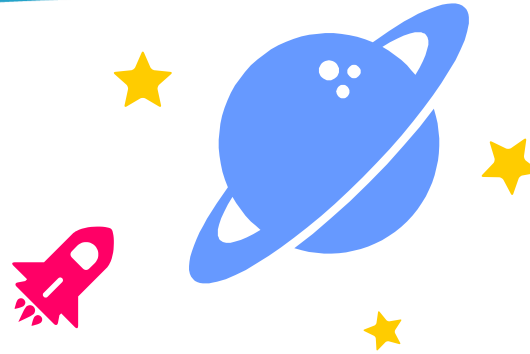
## Dashboard de minikube para kubernetes

\$ minikube dashboard

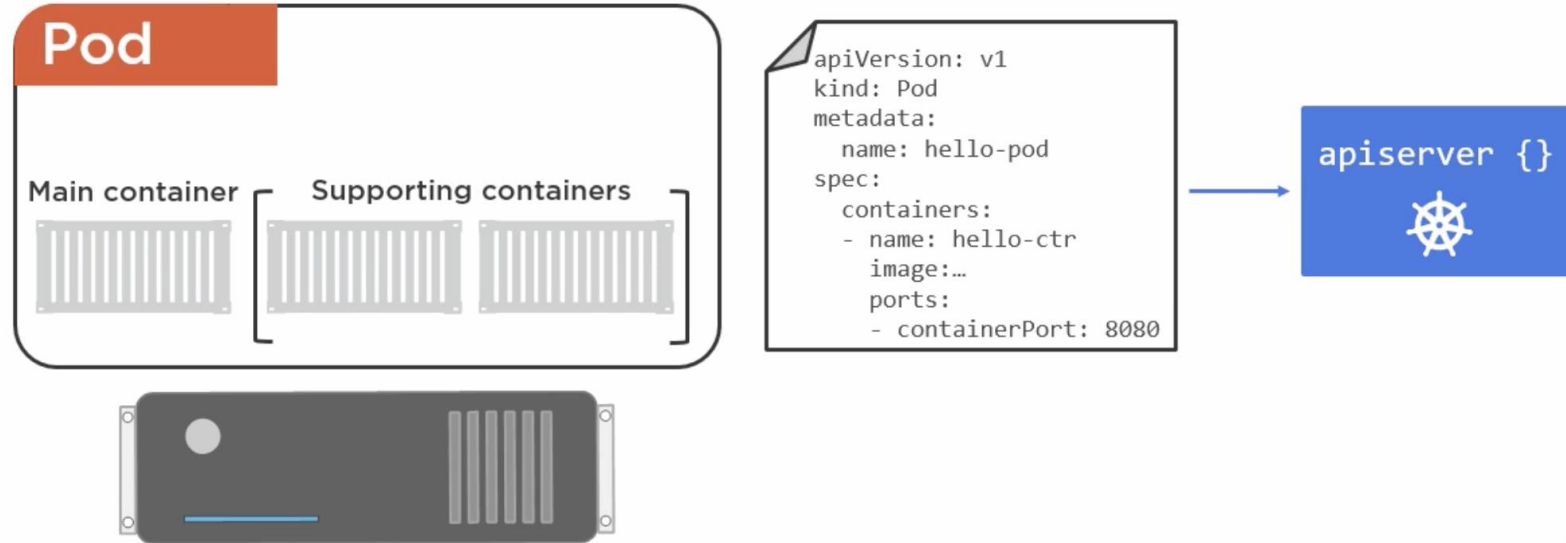
# Lab 12



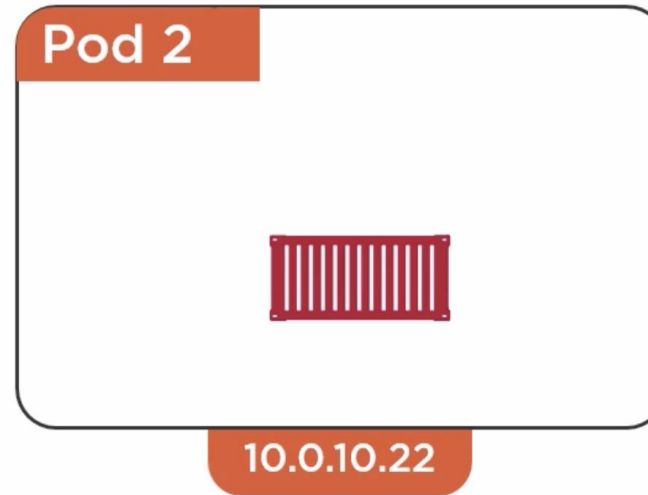
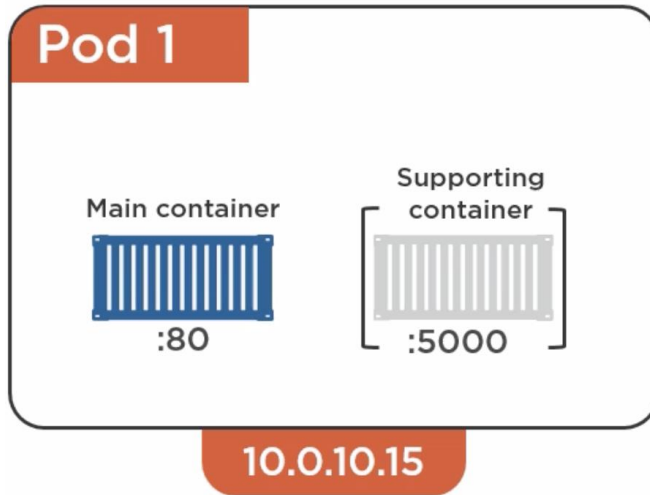
Pods



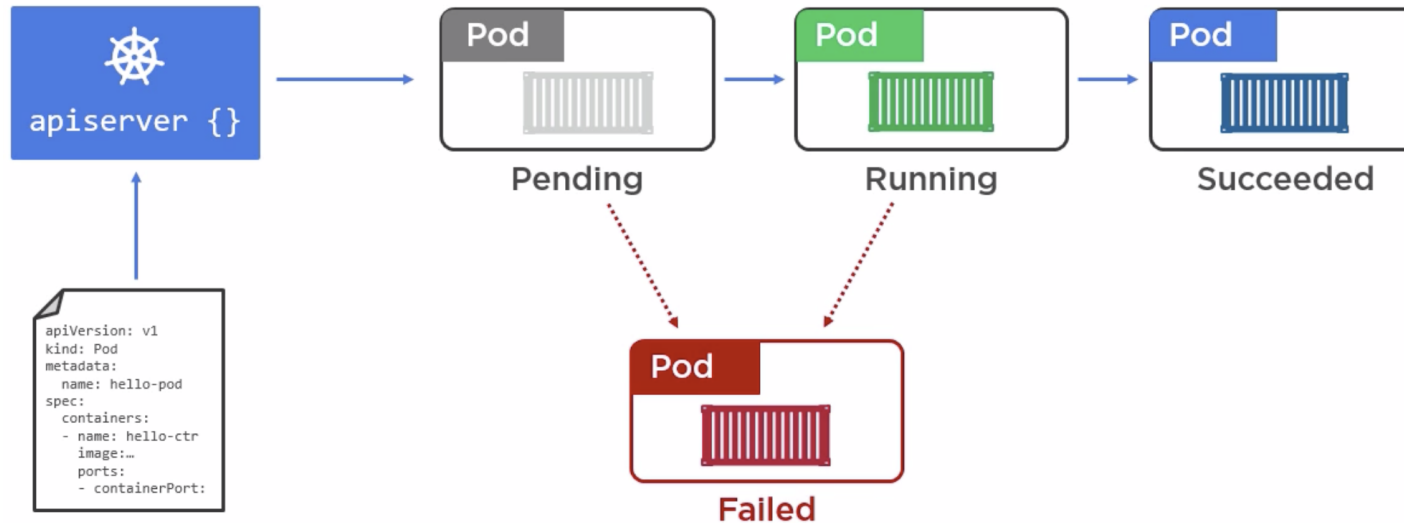
# Pods



# Pods



# Pod lifecycle





# Pods

- > \$ kubectl get pods
- > \$ kubectl get pods --all-namespaces

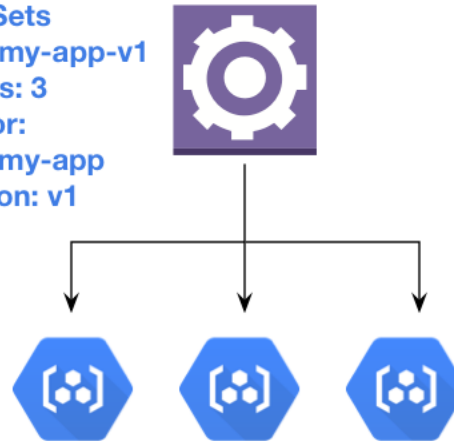
## Manifiesto pod.yml

- > \$ kubectl create -f pod.yml
- > \$ kubectl get pods
- > \$ kubectl describe pods

# Pods Replication controller (desired state)

## ReplicaSets

- name: my-app-v1
- replicas: 3
- selector:
  - app: my-app
  - version: v1



# Pods (Replication controller)

- > \$ kubectl get pods
- > \$ kubectl delete pods hello-pod

Manifiesto rc.yml

- > \$ kubectl create -f rc.yml
- > \$ kubectl get rc
- > \$ kubectl describe rc

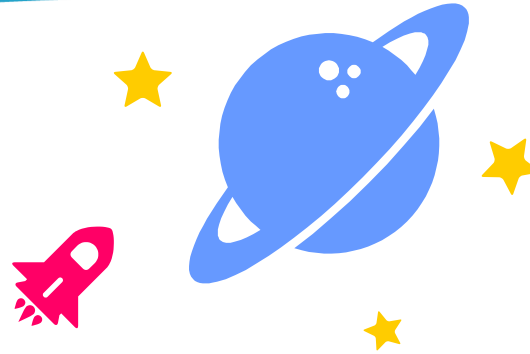
Modificar rc.yml

- > \$ kubectl apply -f rc.yml
- > \$ kubectl get rc -o wide
- > \$ kubectl get pods

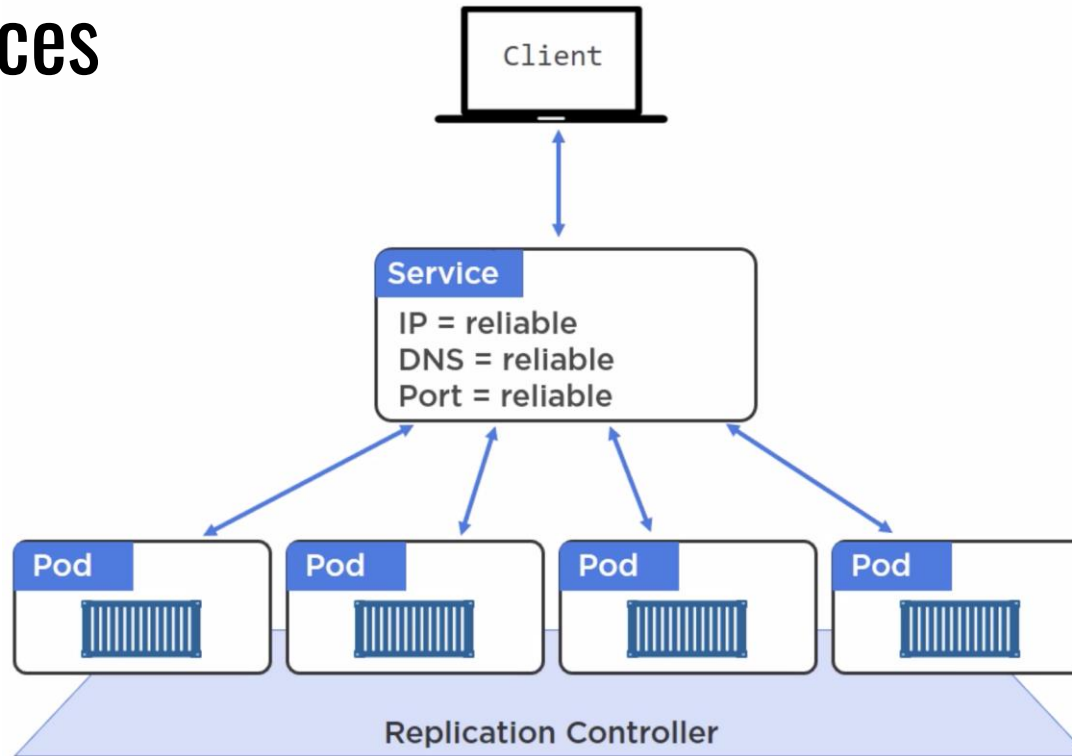
# Lab 13



Services



# Services



# Services

- `$ kubectl get rc`
- `$ kubectl expose rc hello-rc --name=hello-svc --target-port=80 --type=NodePort`
- `$ kubectl describe svc hello-svc`
- `$ minikube ip`
- `$ kubectl delete svc hello-svc`

# Services (declarative yml)

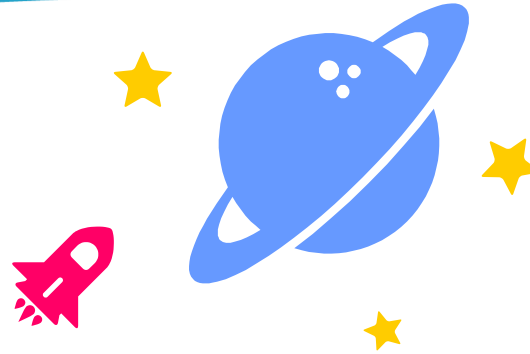
Generar manifiesto svc.yml

- › \$ kubectl describe pods | grep app
- › \$ kubectl create -f svc.yml
- › \$ minikube ip
- › \$ kubectl describe svc hello-svc
- › \$ kubectl get ep
- › \$ kubectl describe ep hello-svc

# Lab 14

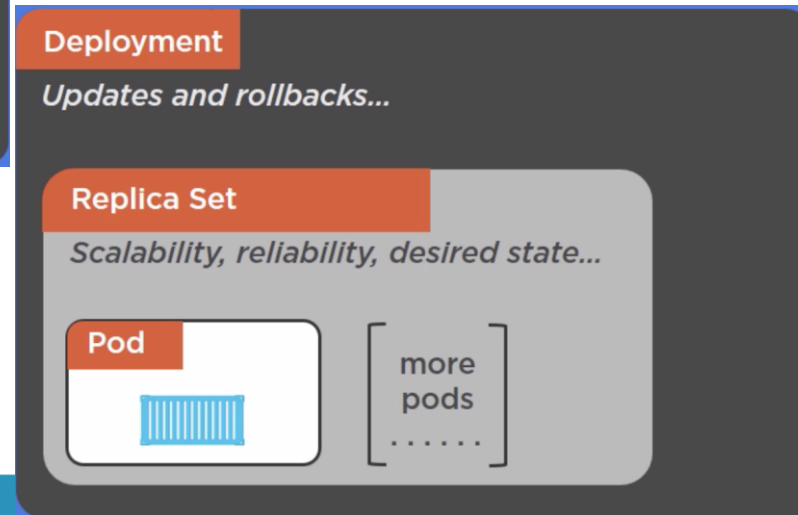
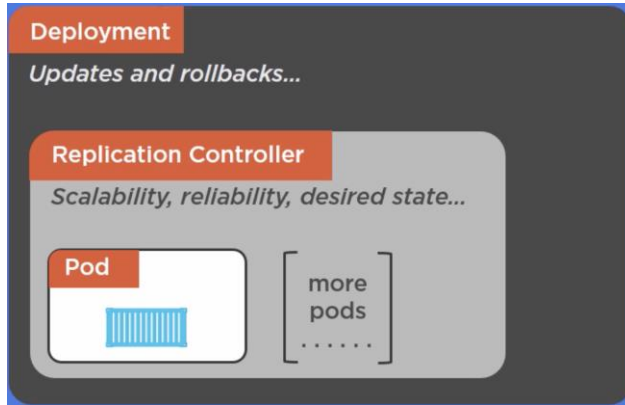


Deployments





# Deployment



# Deployment

- `$ kubectl get rc`
- `$ kubectl delete rc hello-rc`
- `$ kubectl create -f deploy.yml`
- `$ kubectl describe deploy hello-deploy`
- `$ kubectl get rs (replica set)`
- `$ kubectl describe rs`

# Deployment (update)

Modificar yml para agregar nueva imagen docker

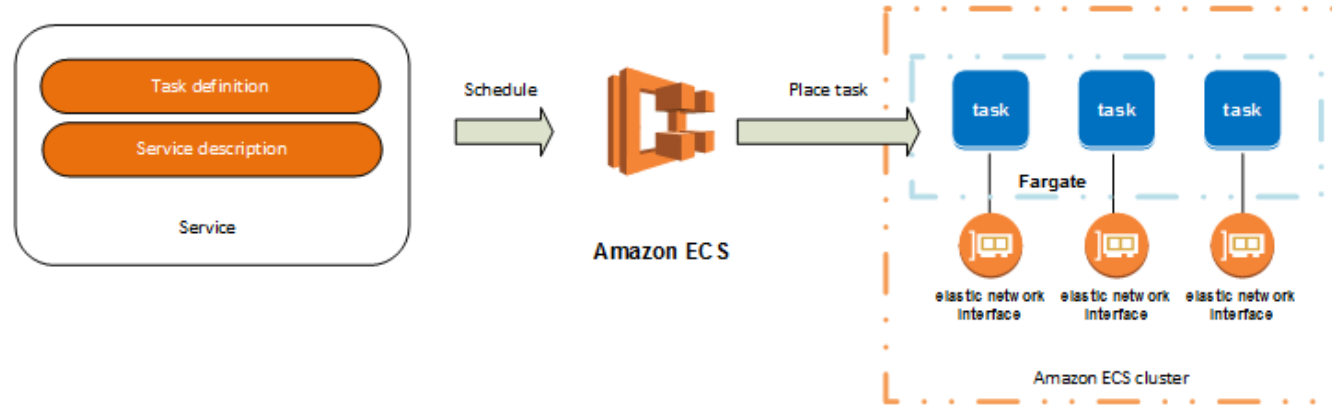
- > \$ kubectl apply -f deploy.yml --record
- > \$ kubectl rollout status deployment hello-deploy
- > \$ kubectl get deploy hello-deploy
- > \$ kubectl rollout history deployment hello-deploy
- > \$ kubectl get rs
- > \$ kubectl describe deploy hello-deploy
- > \$ kubectl rollout undo deployment hello-deploy --to-revision= 1
- > \$ kubectl get deploy
- > \$ kubectl rollout status deployment hello-deploy

# 8. AWS ECS

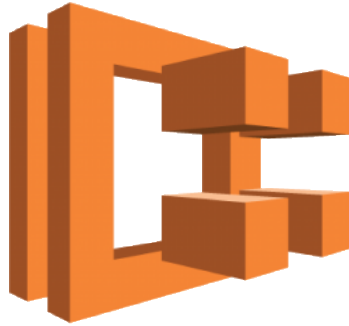
Administrar contenedores en AWS

***“Amazon Elastic Container Service (Amazon ECS) es un servicio de administración de contenedores altamente escalable y rápido que facilita la tarea de ejecutar, detener y administrar contenedores de Docker en un clúster. ”***

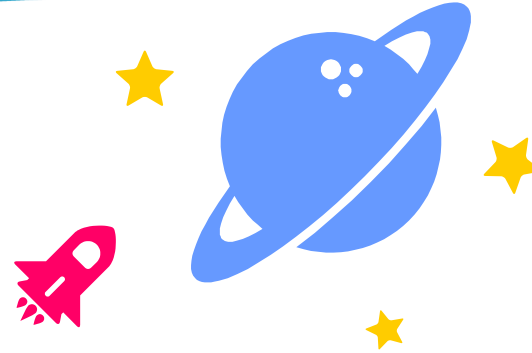
# Arquitectura



# Lab 15



Amazon ECS



Despliegue en AWS ECS

# Configurar cluster

Modificar yml para agregar nueva imagen docker

- > \$ kubectl apply -f deploy.yml --record
- > \$ kubectl rollout status deployment hello-deploy
- > \$ kubectl get deploy hello-deploy
- > \$ kubectl rollout history deployment hello-deploy
- > \$ kubectl get rs
- > \$ kubectl describe deploy hello-deploy
- > \$ kubectl rollout undo deployment hello-deploy --to-revision= 1
- > \$ kubectl get deploy
- > \$ kubectl rollout status deployment hello-deploy





# THANKS!



[www.certificatic.org](http://www.certificatic.org)